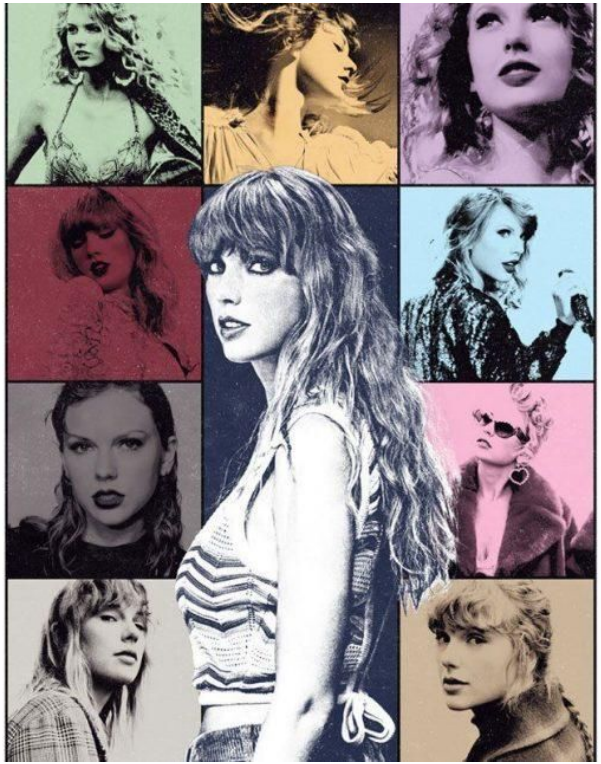
Evaluación

2do cuatri 2023
Paula Pérez Bianchi

DEPARTAMENTO
DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

THE ERAS TOUR ARGENTINA

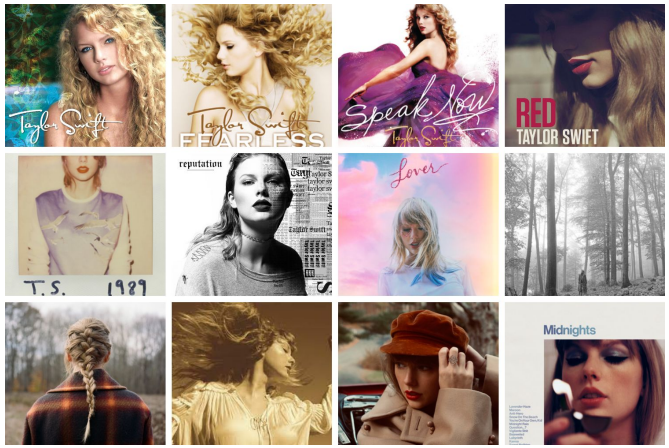


Taylor Swift está realizando un gira mundial donde realiza un recorrido por toda su discografía. En cada concierto Taylor canta dos **canciones sorpresa**. Su equipo tiene la importante tarea de **evitar que se filtren cuales** serán esas canciones. Esto es una tarea muy compleja ya que muchas personas están involucradas en la producción del espectáculo.

En particular es muy importante que los **iluminadores** sepan si las canciones van a ser de alguno de los álbumes **Folklore o Evermore** pues si Taylor canta una canción de estos álbumes ellos deben encender unas **luces especiales**.

TAYLOR SWIFT
THE ERAS TOUR

El equipo de producción de Taylor junta **muchos datos de las canciones** para hacer análisis del mercado.
El equipo cree que la mejor solución al problema es crear un **clasificador de canciones** que dado algunos datos de una canción pueda decir si es de uno de estos álbumes o no.



Luego justo antes de que Taylor cante pueden pasarle al iluminador los datos de la canción y este usará el clasificador para saber si debe prender las luces especiales o no.

Para esto quieren contratar a un equipo de científicos de datos que arme el clasificador.

track_name	danceability	energy	key	liveness	valence	tempo	time_signature	duration_ms	is_folklore_or_evermore
Tim McGraw	0.580	0.491	0.0	0.1210	0.425	76.009	4.0	232107.0	False
Picture To Burn	0.658	0.877	7.0	0.0962	0.821	105.586	4.0	173067.0	False
Teardrops On My Guitar	0.621	0.417	10.0	0.1190	0.289	99.953	4.0	203040.0	False
A Place In This World	0.576	0.777	9.0	0.3200	0.428	115.028	4.0	199200.0	False

OBJETIVO:

Predecir dado los datos de danceability, energy, etci de una canción si es de los discos Evermore o Folklore o es de otro álbum.

Armen un **clasificador** para este problema con alguna de las técnicas que vimos en la materia (10 min)

¿Confían en que su modelo anda bien?

¿Qué modelo le damos al equipo de Taylor?

*¿Se puede **medir** cuán bueno es el modelo en la realidad?*

¿Cómo medimos? ¿Qué métrica usamos?

¿Con qué datos evaluamos?

Si ustedes fueran el equipo de Taylor,
¿Qué harían para ver si el modelo que le entregamos nosotres es bueno?

Metodología de Evaluación

Objetivos:

- ➔ **Comparar dos modelos** para encontrar el mejor (por ej, ¿Uso un árbol de altura 5 o de altura 7?)
- ➔ **Estimar la performance** en la realidad “in the wild”. ¿Mi modelo anda bien para datos nuevos?

Es **fácil errar** en la creación de un modelo (entrenamiento), pero también es **fácil darse cuenta**.

Es **fácil errar** en la evaluación y **no es fácil darse cuenta**.

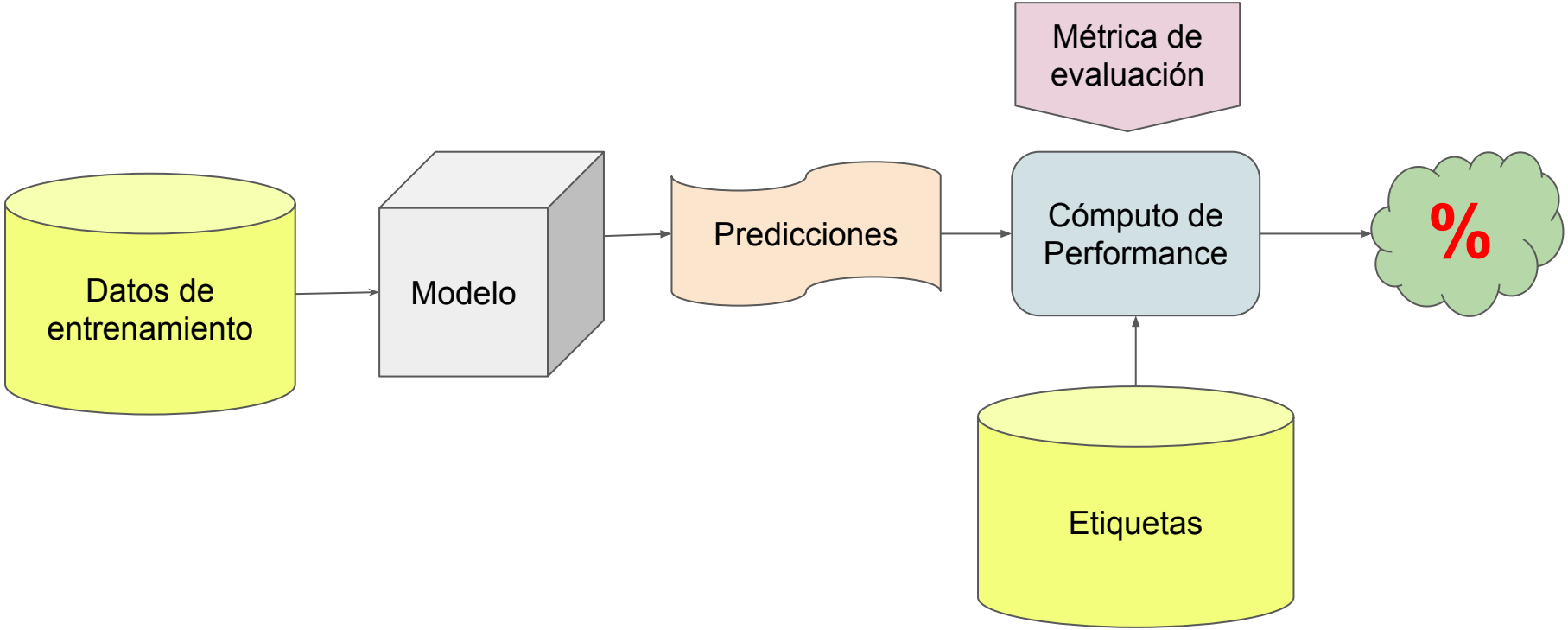
Evaluar bien significa entender bien el **caso de uso**, entender **qué me importa** del problema y qué no, entender **qué métrica/s** refleja/n lo que quiero capturar, entender qué mecanismo de evaluación usar, entender **cómo no hacer/se trampa** como hago que **no me hagan trampa**.

Evaluación de modelos

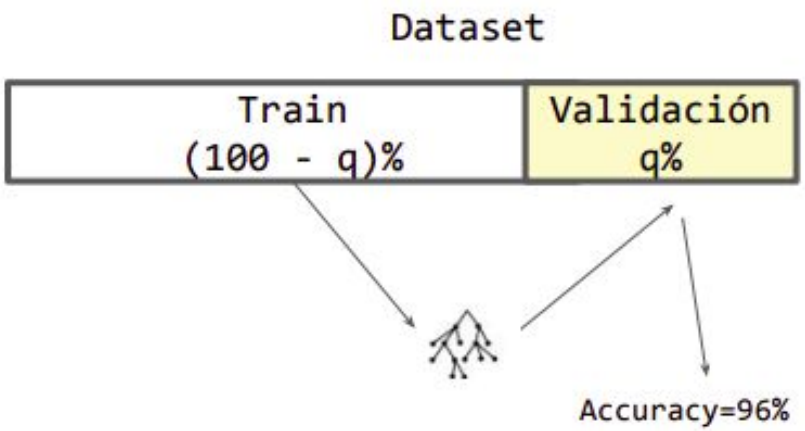
Para un modelo específico (por ej, árbol de decisión de altura 5 con criterio gini)

¿Cuál será la performance de nuestro modelo en la realidad?

Pipeline de Evaluación



Cross Validation (Validación Cruzada)



** Al set de validación también se le llama test.

¿Por qué no medimos sobre los datos de entrenamiento?

1) Separo una porción de los datos que **NO voy a usar para entrenar!**
2) Mido la performance del modelo sobre los datos de validación (en general suelen ser el 5%, 10%, 20% del total de datos)

⚠️ ¿Separo al azar los datos? ¿Qué puede pasar?

→ Esta técnica suele ser suficiente si tengo muchos datos.

`sklearn.model_selection.`**`train_test_split`**(**arrays, test_size=None, train_size=None, random_state=None, shuffle=True, stratify=None*) [source]

Split arrays or matrices into random train and test subsets.

Mezclo los datos antes de separar

shuffle : *bool, default=True*

Whether or not to shuffle the data before splitting. If shuffle=False then stratify must be None.

Establecer una distribución de las clases

stratify : *array-like, default=None*

If not None, data is split in a stratified fashion, using this as the class labels. Read more in the User Guide.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test =
            train_test_split(df, labels, test_size = 0.1, random_state= 7,
        stratify= labels)
```

K-Fold Cross Validation

¿Y si tenemos mala suerte al separar los datos para entrenamiento/validación?

- La estimación de performance del modelo podría no ser realista. En especial cuando tenemos pocos datos!

La idea es generar distintas particiones para armar k datasets distintos a partir de mi dataset original.

Acá estoy evaluando un **modelo particular con sus hiper-parámetros**. Por ejemplo, un árbol de decisión con altura 10 o un modelo de KNN con k = 3.

	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 1	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 2	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 3	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 4	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
Split 5	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5

Input:

L (un algoritmo de aprendizaje + sus hiperparametros)

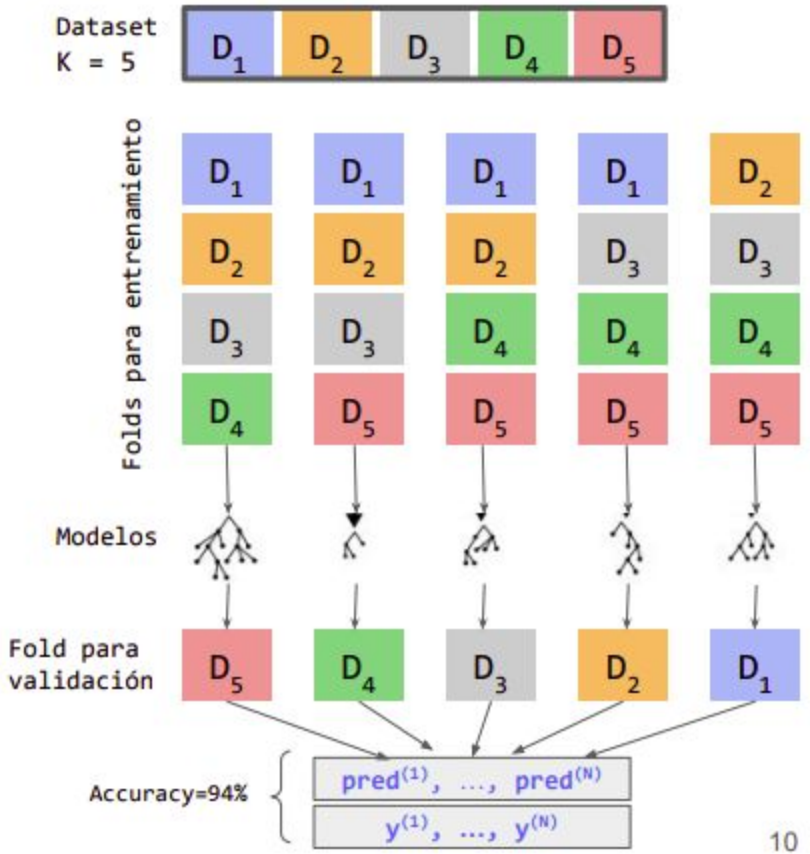
D el "dataset'

1. Separamos **D** en **K** subconjuntos a los que llamamos folds: **D1 , D2 , D3 , …, DK**

2. Construimos **K modelos** con el algoritmo y hiperparametros de **L** que serán entrenados utilizando todos los datos salvo los del k-ésimo fold.

3. Para cada **x(i) ∈ D**:
pred(i) = modelo_que_no_vio_ese_dato_en entrenamiento.predict()
predicciones[i] = pred(i) # juntamos las predicciones como si vinieran todas del mismo modelo

4. Computamos alguna **métrica del error(**)** sobre el conjunto entero **predicciones** vs **y**



(**) Una variación muy usada es medir el error sobre el fold de validación y luego promediar el valor de todos los folds.

Algunas consideraciones

¿Cómo separo los datos en cada fold? ¿Al azar?

¿Las clases están desbalanceadas?
¿Qué sucede con instancias **no independientes**?

Ej: En reconocimiento del habla, un mismo hablante no debe aparecer en ambos conjuntos. (¿O si?)

CLAVE: No quiero en el conjunto de validación un dato que se usó para entrenar el modelo.
(¿Qué pasa si es muy parecido? ¿Y si comparte una propiedad muy fuerte con otros datos?)

No perder de vista lo importante, ¿Qué queremos que aprenda el modelo?
(ej, perros y árboles, huskies vs lobos)

`sklearn.model_selection.cross_val_score(estimator, X, y=None, *, groups=None, scoring=None, cv=None, n_jobs=None, verbose=0, fit_params=None, pre_dispatch='2*n_jobs', error_score=nan)` [source]

```
cross_val_score(taylor_clf, X, y, cv=5)
```

Calcula la performance sobre cada fold usando un modelo entrenado con todos los folds menos ese. ¿Qué nos devuelve esto?

`class sklearn.model_selection.KFold(n_splits=5, *, shuffle=False, random_state=None)`

```
from sklearn.model_selection import KFold
kf = KFold(n_splits=2, shuffle=True)
list(kf.split(X, y))
```

Separa los datos en k-folds consecutivos

`class sklearn.model_selection.StratifiedKFold(n_splits=5, *, shuffle=False, random_state=None)`

Separa los datos en k-folds preservando la distribución de las clases

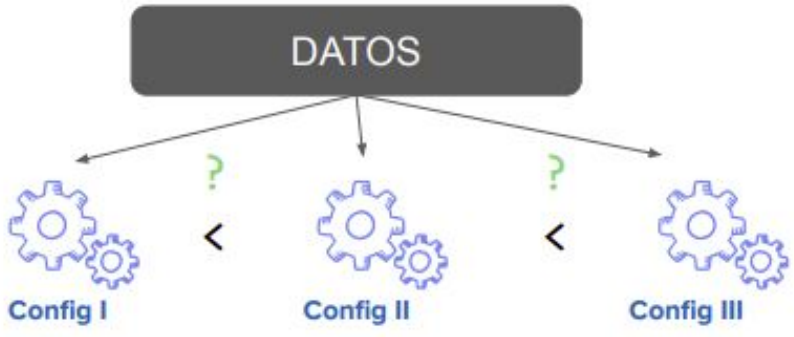
Ejercicio para el hogar: *Utilizar alguno de estos métodos para separar los datos y calcular la performance de su clasificador para Taylor Swift usando el algoritmo anterior.*


I UNDERSTAND SO LET'S TAKE A BREAK
AND HAVE SOME TEA
!!!!
makeameme.org

Búsqueda del mejor modelo

Para poder elegir un **modelo final**, es natural explorar distintas combinaciones de **hiper-parámetros** y distintos **algoritmos de aprendizaje.**

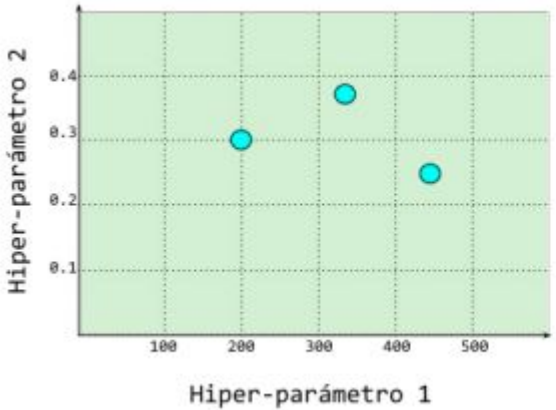
Por ejemplo, ¿Uso un árbol de altura 5 o uno de altura 10 o KNN con k = 6?



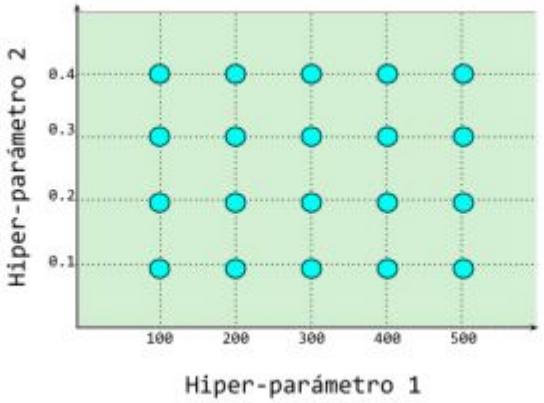
Selección de Modelos

¿Cómo elegirían cuáles son los mejores hiperparametros para los modelos?

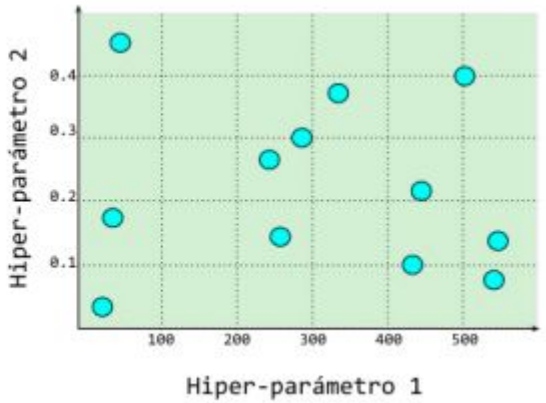
Manual Search
Setear a mano hiper-parámetros que pensamos que van a funcionar bien.



Grid Search
Plantear opciones y explorar todas las combinaciones



Random Search
Se plantean distintas opciones y se exploran combinaciones al azar.



class `sklearn.model_selection.`**`GridSearchCV`**(*estimator, param_grid, *, scoring=None, n_jobs=None, refit=True, cv=None, verbose=0, pre_dispatch='2*n_jobs', error_score=nan, return_train_score=False*) [source]

Para cada combinación de hiper parámetros, la evalua haciendo cross-validation

class `sklearn.model_selection.`**`RandomizedSearchCV`**(*estimator, param_distributions, *, n_iter=10, scoring=None, n_jobs=None, refit=True, cv=None, verbose=0, pre_dispatch='2*n_jobs', random_state=None, error_score=nan, return_train_score=False*)

Toma n_iter combinaciones de hiper parámetros al azar y las evalua haciendo cross-validation

```
from sklearn.model_selection import RandomizedSearchCV
hyper_params = {'criterion' : ["gini", "entropy"],
                'max_depth' : [2,3,4,5] }
clf = RandomizedSearchCV(tree, hyper_params, random_state=0)
search = clf.fit(X, y)
search.best_params_
search.best_score_
```

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html#sklearn.model_selection.RandomizedSearchCV

Proceso de selección de modelos

1) Armo una **grilla con los hiperparametros** a explorar para cada modelo.

2) **Evaluar** cada configuración con **cross-validation**

3) Tomo como **modelo final** el que obtuvo mejor score en cross-validation

Veamos esto para los *datos de las canciones de Taylor Swift*,

¿Cuál terminó siendo el mejor modelo?
¿Lo habíamos considerado inicialmente?

Si viene Taylor y me pide el score de mi modelo ¿Cuál es el score que le digo?

Held out method

¿La performance que sale de elegir al modelo sirve para ver cómo va a andar el modelo en la realidad?

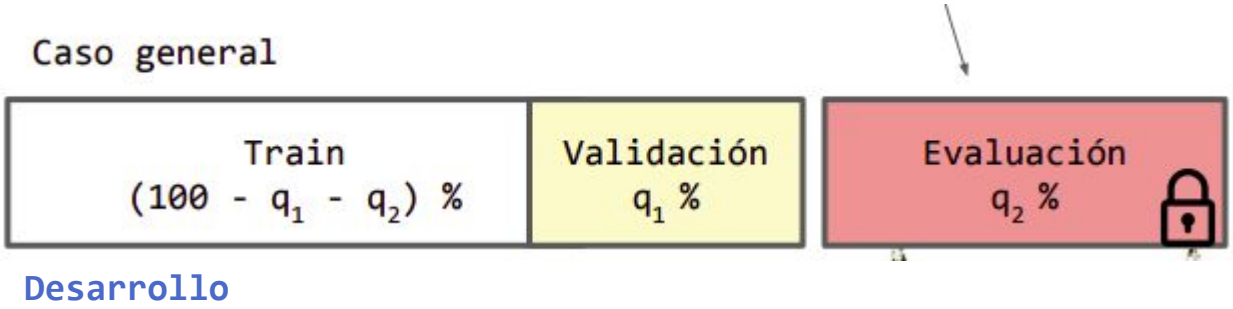
→ Estamos evaluando el modelo en datos que usamos para construirlo!

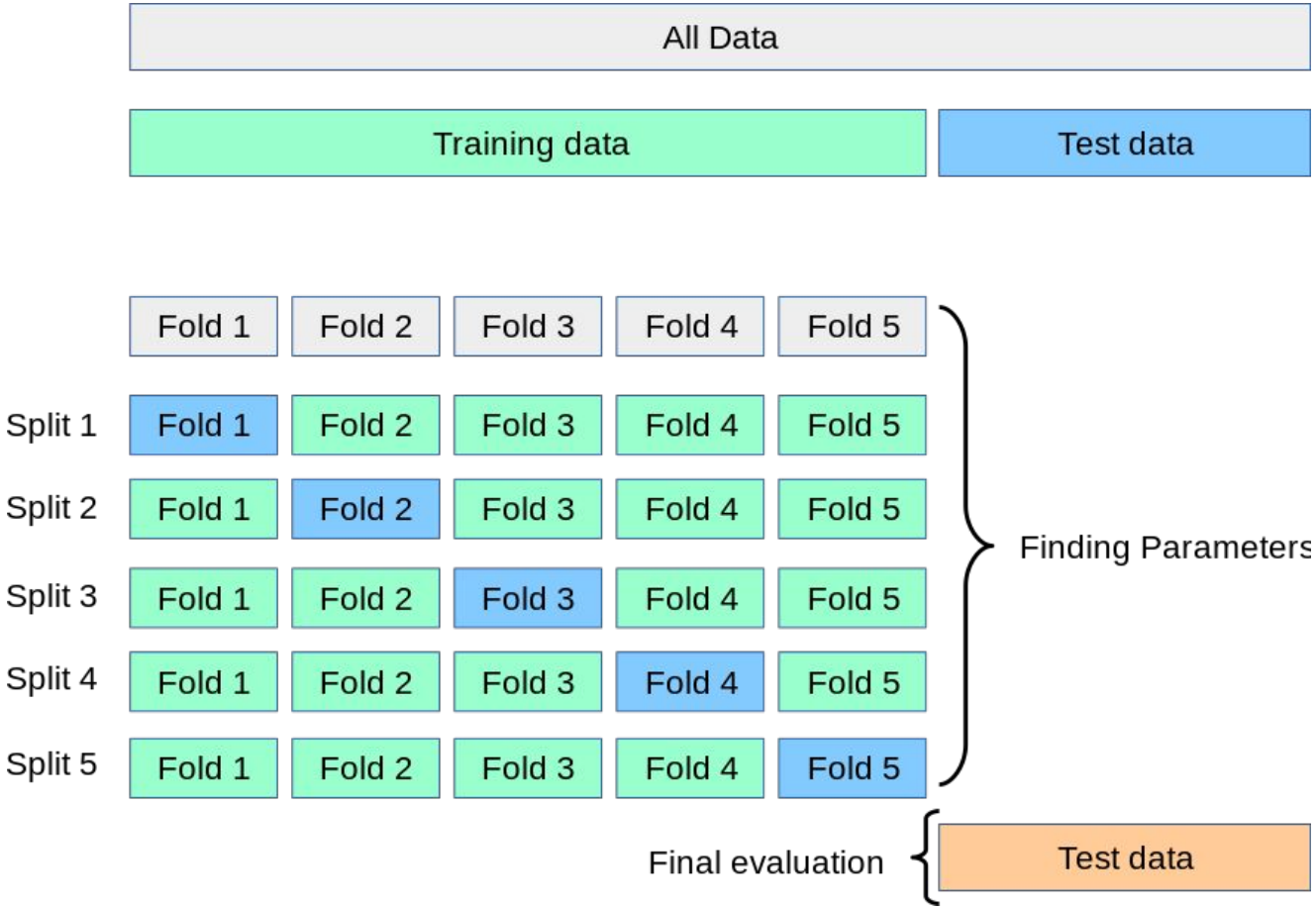
Estamos sobreestimando la performance!

***Solución:* Separo en datos de Desarrollo y Evaluación**

Dev set : *Lo utilizo para seleccionar y entrenar el modelo.*

Eval set : *No lo uso hasta el final! Lo utilizo para hacer la evaluación final sobre el mejor modelo. → Sobre estos datos mido la performance real y la reporto*




All Data
Training data
Test data
Fold 1
Fold 2
Fold 3
Fold 4
Fold 5
Split 1
Split 2
Split 3
Split 4
Split 5
Finding Parameters
Final evaluation
Test data

Proceso completo

1) Separo los datos en **DEV** y **EVAL**
2) Con los datos en **DEV**

   a) Armo una **grilla con los hiperparametros** a explorar para cada modelo.

   b) **Evaluar** cada configuración con **cross-validation**

   c) Tomo como **modelo final** el que obtuvo mejor score en cross-validation

3) Con los datos en **EVAL**

   a) Calculo la performance del modelo. **¡No vuelvo atrás!**

Métricas de performance

La noción de equivocarse,
¿Es equivalente para clasificación y regresión?

¿Hay errores más graves que otros?

Métricas de performance

Regresión→ MSE

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

Nos interesa cuantificar el error. ¿Por cuánto le erro?

Clasificación → Accuracy

$$Accuracy = \frac{|\text{Instancias bien clasificadas}|}{|\text{total de instancias}|}$$

¿Qué significa si me dicen: Mi modelo tiene accuracy del 80%?

→ De 100 muestras 80 las clasificó bien

⚠️ No dice nada sobre el **tipo de aciertos y errores** que comete el modelo.

Tipos de Error

Falso Positivo : Era negativo pero dijimos positivo!

Falso Negativo: Era positivo pero dijimos negativo!

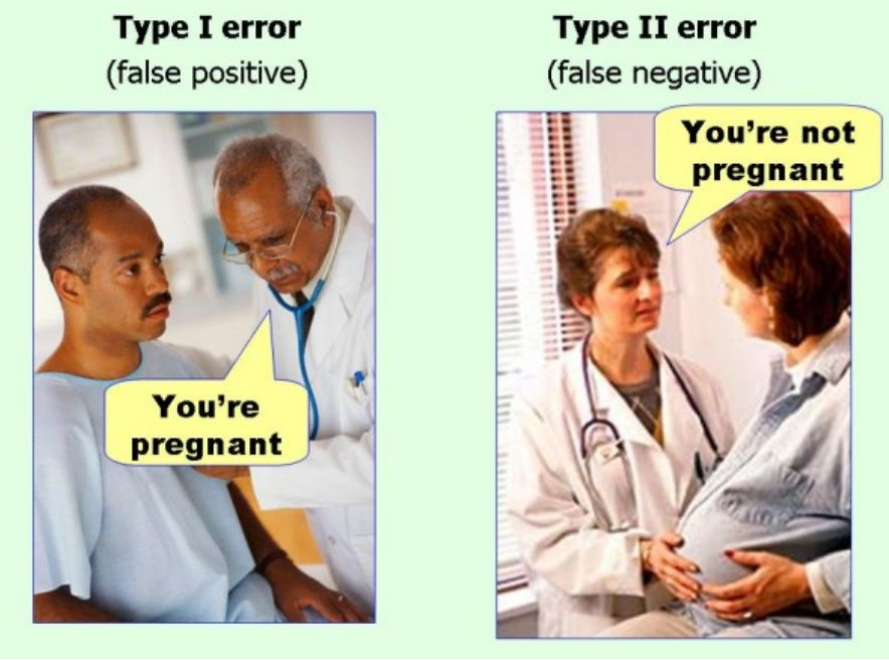
EJERCICIO (5min)

Identifiquen el tipo de error que se comete en:

→ Un filtro de spam que marca todos los emails como spam.

→ Un detección de fraude que siempre dice “no fraude”.

→ Un Identificación de meteoritos que siempre dice “NO”.



*Nos interesa a continuación **medir mejor** (con más detalle) y poder asignar costos a cada tipo de error*

Matriz de confusión (clasificación binaria)

		Predicted	
		Negative **(N)** **-**	Positive **(P)** **+**
Actual	Negative **-**	True Negative **(TN)**	**False Positive (FP)** **Type I Error**
	Positive **+**	**False Negative (FN)** **Type II Error**	True Positive **(TP)**

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

¿Les recuerda a algo esta tabla?

Precisión y Recall

$$\text{Precisión} = \frac{TP}{TP + FP}$$

De las instancias clasificadas como positivas, cuántas lo son.
(cuán útiles son los resultados de búsqueda)

$$\text{Recall} = \frac{TP}{TP + FN}$$

De las instancias positivas, cuántas fueron clasificadas como positivas
(cuán completos son los resultados)

EJERCICIO ¿Cuál medida (**precisión**/**recall**) debería priorizar cada uno de estos sistemas? (5min)
1. Filtro de spam: que descartará directamente los emails sospechosos.
2. Detección de fraude: prepara un listado de casos sospechosos para ser revisados por humanos.

Caso binario: F-score

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

Vemos que alto recall y alto precision son dos características deseables del sistema.
Pero un sistema con recall 100% y precision 0% o viceversa, no es un buen sistema.

Curvas ROC

Gráfico Recall vs. False Positive Recall.
Me permiten comparar entre distintos clasificadores con distinta confusion matrix.

$$\text{Recall} = TPR = \frac{TP}{TP + FN}$$

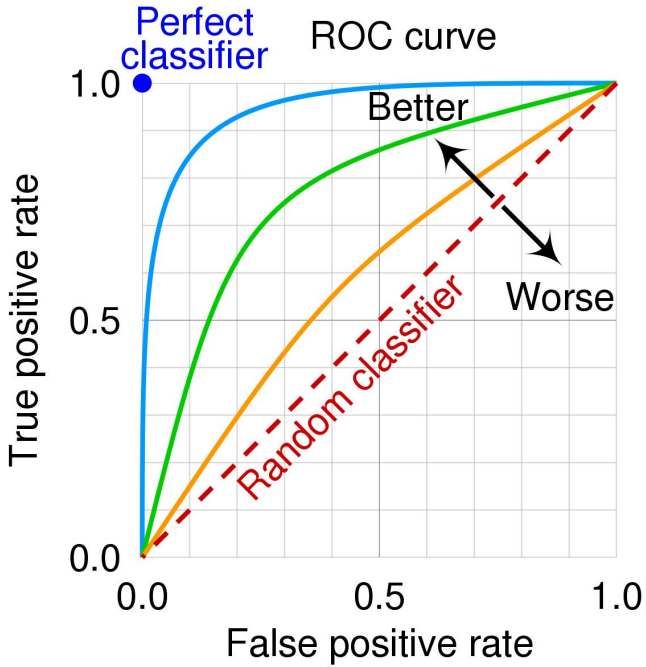
Cuantas instancias positivas son correctamente clasificadas

$$\text{FPR} = \frac{FP}{FP + TN}$$

vs.

Cuantas instancias negativas fueron clasificadas como positivas

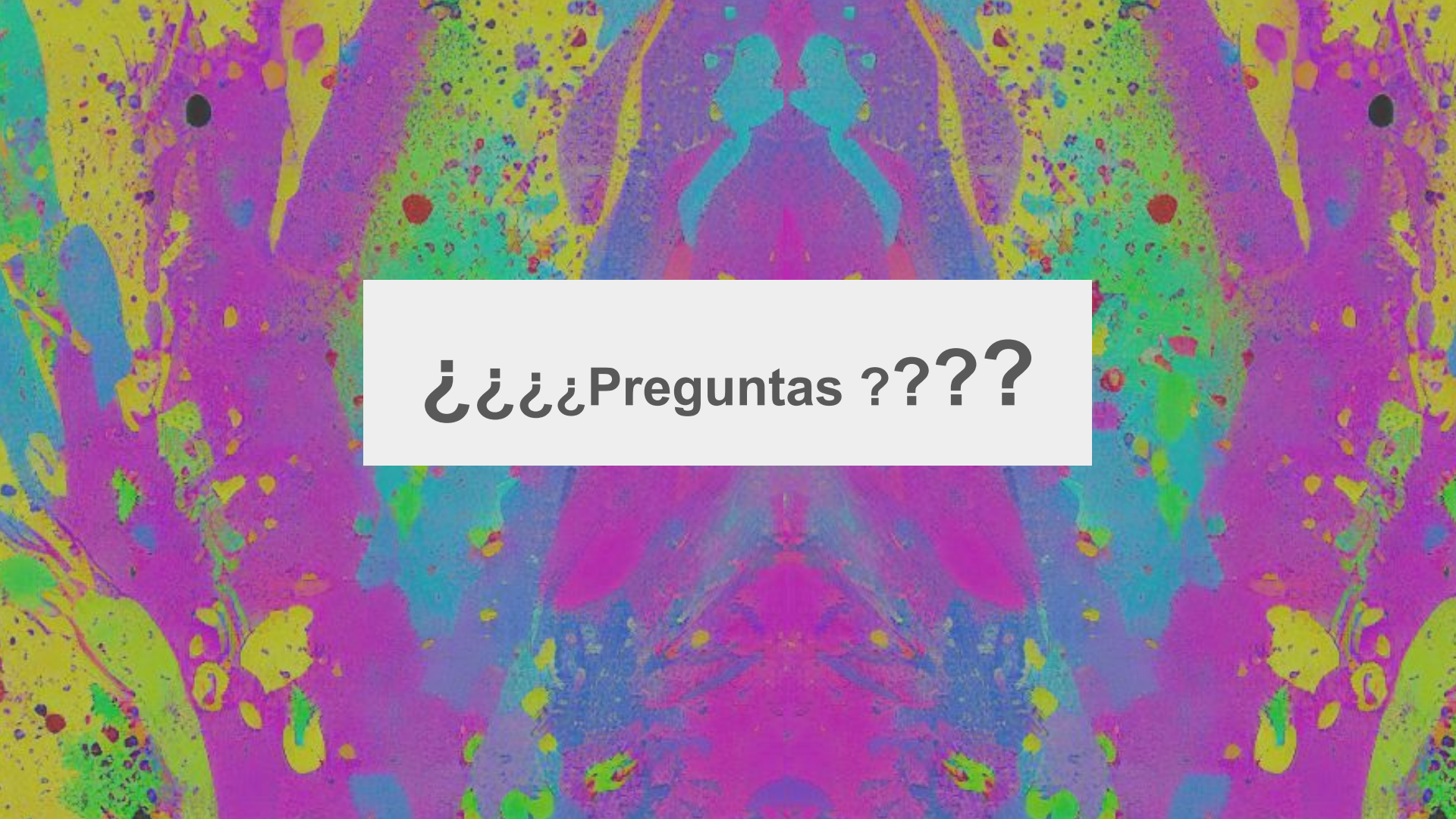
FPR = 1-Precisión



https://www.youtube.com/watch?v=4jRBRDbJemM

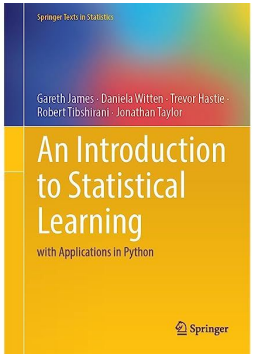
Resumen

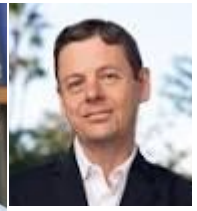
- ★ Evaluación vs. Selección de modelos
- ★ Evaluación cruzada
- ★ Cross-validation
- ★ Exploración de hiper parámetros
- ★ Metodología de evaluación
- ★ Hold-out set
- ★ Métricas: Precisión y Recall


¿¿¿¿Preguntas ????

Bibliografía + Créditos

- Müller & Guido, "Introduction to Machine Learning with Python", O'Reilly, 2016. Código. Capítulo 5
- James, Witten, Hastie & Tibshirani, "An Introduction to Statistical Learning with Applications in R", 6th ed, Springer, 2015. Capítulo 5.1 (CV).

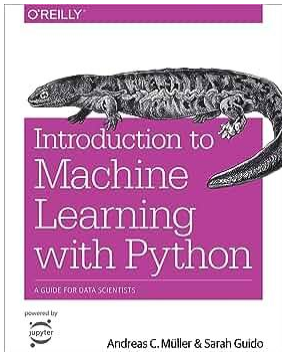




Otros recursos

https://www.youtube.com/watch?v=fSytzGwwBVw

https://www.youtube.com/watch?v=Kdsp6soqA7o



Créditos

- Clases de la materia de AA dictada por Pablo Brusco en 1c2023
- Clase correspondiente a este tema del cuatri anterior dictada por Manuela Cedeiro
- Autores de los magníficos memes e imágenes.